NAVAL
POSTGRADUATE
SCHOOL

MONTEREY, CALIFORNIA

THESIS

DATABASE SYSTEM DESIGN AND IMPLEMENTATION FOR MARINE AIR-TRAFFIC-CONTROLLER TRAINING

by

Anthony J. Ambriz

June 2017

Thesis Advisor:	Neil C. Rowe
Second Reader:	Arijit Das

Approved for public release. Distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.

1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** June 2017	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE DATABASE SYSTEM DESIGN AND IMPLEMENTATION FOR MARINE AIR-TRAFFIC-CONTROLLER TRAINING	**5. FUNDING NUMBERS**
6. AUTHOR(S) Anthony J. Ambriz	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB number ____N/A____.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.	**12b. DISTRIBUTION CODE**

13. ABSTRACT (maximum 200 words)

This project focused on the design, development, and implementation of a centralized electronic database for air-traffic-controller (ATC) training within the Marine Corps. The Marine Corps currently has individual electronic training database-management systems at the unit level. The existing training database-management systems are client based, unstandardized, and have significant data-entry redundancy. We developed a centralized system that is scalable and capable of supporting detailed training statistics. We identified several critical features required for successful implementation and developed a system capable of importing and processing existing training data.

We analyzed an existing training database-management system used at the unit level and developed methods for accurately exporting the existing system's stored generic data. We processed the raw data and properly migrated the data to a prototype database-management system, which involved processing for proper data integrity, database normalization, and enforcing unique record-identifier usage. We developed a front-end graphical user interface to interact with the developed system and the stored training data. Future system improvements include migration to a military network, adaptation by additional units, and full integration with the Navy's Visual Information Display System.

14. SUBJECT TERMS Marine Corps, ATC, Air Traffic Control, MACCS, TMS, training database-management system, T&R, training and readiness, paperless records, MPR, MACCS Performance Records, databases, MOS	**15. NUMBER OF PAGES** 73
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89) Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

DATABASE SYSTEM DESIGN AND IMPLEMENTATION FOR MARINE AIR-TRAFFIC-CONTROLLER TRAINING

Anthony J. Ambriz
Captain, United States Marine Corps
B.S., Drexel University, 2008

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
June 2017

Approved by: Neil C. Rowe
Thesis Advisor

Arijit Das
Second Reader

Peter J. Denning
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This project focused on the design, development, and implementation of a centralized electronic database for air-traffic-controller (ATC) training within the Marine Corps. The Marine Corps currently has individual electronic training database-management systems at the unit level. The existing training database-management systems are client based, unstandardized, and have significant data-entry redundancy. We developed a centralized system that is scalable and capable of supporting detailed training statistics. We identified several critical features required for successful implementation and developed a system capable of importing and processing existing training data.

We analyzed an existing training database-management system used at the unit level and developed methods for accurately exporting the existing system's stored generic data. We processed the raw data and properly migrated the data to a prototype database-management system, which involved processing for proper data integrity, database normalization, and enforcing unique record-identifier usage. We developed a front-end graphical user interface to interact with the developed system and the stored training data. Future system improvements include migration to a military network, adaptation by additional units, and full integration with the Navy's Visual Information Display System.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

I. INTRODUCTION 1
  A. STANDARDIZATION ISSUES 1
  B. INSTITUTING STANDARDIZATION 2
  C. TANGIBLE BENEFITS 3
  D. THESIS ORGANIZATION 4

II. PREVIOUS ATTEMPTS AT IMPLEMENTING TRAINING-MANAGEMENT SYSTEMS 5
  A. ATTEMPTED SOLUTIONS 5
    1. Marine-Sierra Hotel Aviation Readiness Program 5
    2. Local Databases 9
  B. SIMILAR APPLICATIONS 10

III. APPLICATION DESCRIPTION 13
  A. THE CURRENT TRAINING-RECORD SITUATION 13
    1. Workload Duplication 14
    2. Qualification Timeline Training 15
    3. Report Generation 18
    4. Data Validation 20
  B. ASSUMPTIONS MADE FOR A NEW SYSTEM 24

IV. PROGRAM DESCRIPTION 27
  A. HARDWARE/SOFTWARE ENVIRONMENT SPECIFICATIONS 27
  B. SAMPLE DATABASE ANALYSIS 27
  C. SYSTEM DEVELOPMENT 30
  D. SAMPLE DATA EXPORTATION 32
  E. DATA IMPORTATION 36
  F. DATA PROCESSING 37
  G. END-USER REPORTS 38

V. RESULTS DISCUSSION 41
  A. DATA EXPORTING 41
  B. DATA IMPORTING 41
  C. FRONT-END GRAPHICAL USER INTERFACE TESTING 42

VI. CONCLUSIONS AND FUTURE WORK 43

A. CONCLUSIONS 43
B. FUTURE WORK 44

APPENDIX A. EXAMPLE SQL*LOADER FILES 47
A. .CTL CONTROL FILE 47
B. .SH BATCH FILE 47
C. LOG OUTPUT FILES 48
1. .BAD Output File 48
2. .LOG Output File 48

APPENDIX B. ORIGINAL AND GENERATED MPR DOCUMENTS 51

LIST OF REFERENCES 53

INITIAL DISTRIBUTION LIST 55

LIST OF FIGURES

Figure 1. Processing Exported Data 33

Figure 2. Report Generation 39

Figure 3. Original MPR Audit Report Record. Source: Department of Defense (2016). 51

Figure 4. Front-End Display of Generated MPR Audit Report Record 52

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

Table 1. Training Information Stored within M-SHARP for a Training and Readiness Event. Source: InnovaSystems International (2016). 7

Table 2. Possible Qualification Levels within M-SHARP. Source: InnovaSystems International (2016). 8

Table 3. Sample Fields within Training Position Qualification Table 30

Table 4. Oracle Reserved Words. Source: Portfolio et al. (1996, p. B-2). 35

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

ACA1	air-traffic-controller basic course
ATC	air-traffic-control
EDIPI	Electronic Data Interchange Personal Identifier
MACCS	Marine Air Command and Control System
MOS	Military Occupational Specialty
MPR	MACCS Performance Record
PII	personally identifiable information
SQL	Structured Query Language

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to express my gratitude to Professor Rowe and Professor Das for their support and guidance. I would also like to thank the Air Traffic Control unit at Marine Corps Air Station Beaufort for providing valuable data to analyze and process. Lastly, I am especially thankful to my wife, Taylore, for her love, support, and patience.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

The Marine air-traffic-control (ATC) training program documents the qualifications, designations, training progression, and current controller skills of both Marine and civilian air-traffic-controllers within a Marine unit. Although the Department of the Navy has policies that state controller training data documentation requirements, a standard does not exist on how the individual training data must be stored at each unit (Department of the Navy, 2012a). Without these standards, individual units have developed inconsistent local policies for processing, maintaining, and analyzing training data.

A. STANDARDIZATION ISSUES

Münstermann, Weitzel, and Eckhardt (2010) identify that lack of a single centralized training database-management system increases costs and processing time, while reducing data analysis capabilities at a granular level. With separate systems processing training data in each unit, there is a significant amount of redundant data entry as individuals transfer between units. Having no clear personnel tracking guidelines established, units have developed varied methods of identifying individuals within training systems.

Current systems are not easily adaptable to new compliance directives such as removing personally identifiable information (PII) from military records. The Department of Defense (2012) stated units were required to remove social security numbers from any existing database-management systems as part of updated data security protocols. Units were currently social security numbers as a primary means of identifying individuals. Units transitioned to the Electronic Data Interchange Personal Identifier (EDIPI) for identifying individuals. Unfortunately, units lost significant capabilities in analyzing historical data unless the member was still present within the unit, as there lacked a means of correlating a departed member's social security number to the new EDIPI.

The Department of Defense established the EDIPI for identification purposes in 2004 as a replacement to using a member's social security number. An EDIPI comprises 10 digits, with the first nine digits being a unique identifier and the last digit a checksum for the preceding nine digits. The Department of Defense assigns an EDIPI to every member within the Defense Enrollment Eligibility Reporting System, which includes active-duty military, reservists and active-reservists, individual ready reservists, dependents, retirees, contractors, and government civilians.

An EDIPI belongs to one individual for the duration of their life and the EDIPI is never re-assigned. The use of an EDIPI is restricted for Department of Defense business purposes only and has restricted use outside of the Department of Defense. A Memorandum of Understanding is required between the Department of Defense and an outside agency prior to outside agency using the EDIPI. The Department of Defense considers EDIPI exposure low-risk for fraud or identity theft and a PII breach report is not required as a result from exposed EDIPI unless the other PII identifiers are exposed at the same time (Department of Defense, 2012).

Incorrectly modifying training database-management systems can result in data corruption or data loss. As each unit independently developed their training platforms, any system documentation was ad hoc and incapable of use on other systems. Training Chiefs stationed at a unit are continuously required to monitor existing systems for flaws or errors and their resident system knowledge can be lost over time.

B. INSTITUTING STANDARDIZATION

Significant difficulties can occur when establishing a standardized database-management system used across several sub-organizations. New system developers must decide to either develop a database-management system that will overlay the existing training system or develop a new database-management system. Cargill (2011) points out that standardization, while keeping established individually developed systems, forces developers to accept any developed issues within each unit or the individual training database-management systems can lose functionality. Proper conceptualization, development, and implementation are required for successful system implementation.

For every sub-organization, system developers must analyze all processes used to develop a method for implementing data, normalizing it, and developing a streamlined workflow. Documenting every sub-organization's processes is time-consuming, and organizations must be diligent in their preparation. Developers must analyze every sub-organization workflow to ensure no historical data will be lost in the system migration.

An additional issue to consider is the database-management system's practicality. Siha and Saad (2008) suggest that the developed system must be economical, easy to understand, reliable, and adaptable to future changes. Users must be given adequate time to train on the new database-management system. Cargill (2011) says that if organizations provide users inadequate training or developers built the replacement system without end-user input, the database-management system runs a high risk of being used improperly and eventually discarded. The database-management system can also suffer from “feature creep,” or incompatible implementation, resulting in the developed training system not used properly, if at all.

C. TANGIBLE BENEFITS

Cargill (2011) found that when an organization properly standardizes their database-management systems, it establishes a flexible foundation for future design changes, whether due to system upgrades or for regulatory and compliance requirements. Siha and Saad (2008) suggest that organizations can more easily identify critical-failure areas and areas to improve efficiency when processes are accurately mapped and clear workflows identified. Organizations can often eliminate a significant amount of redundant data entry and greatly increase resource utilization, as sub-organizations will no longer be required to manually transpose an individual's administrative data every time they transfer between sub-organizations; currently Marines must carry a hard copy of their training folder when they move to a new organization. This training folder must be re-entered upon ever move between units. The developed database-management system can update data for use by the entire organization.

When organizations establish a single training interface and information workflow, individuals are also no longer required to learn a new database-management system at each sub-organization. Instead, individuals can remain proficient throughout their careers and largely eliminate the time to learn to use a new database-management system upon transfer. Münstermann et al. (2010) concluded that an organization's maintenance costs decrease and overall system quality is improved when a standardized database-management system is implemented. Additionally, when organizations establish a single standardized workflow for every process, they greatly reduce the probability of end-users making mistakes when working with data.

D. THESIS ORGANIZATION

We organized the thesis into six chapters. The second chapter identifies previous attempts to create and consolidate database-management systems. The third chapter details the research methodology and boundaries established when building the designed database-management system. The fourth chapter provides a detailed description of the developed training database-management system, while the fifth chapter discusses the developed system's results. Finally, the sixth chapter provides conclusions and describes paths for future work. Within the appendices are sample Structured Query Language (SQL)*Loader files and forms used for report generation in the developed training database-management system.

II. PREVIOUS ATTEMPTS AT IMPLEMENTING TRAINING-MANAGEMENT SYSTEMS

Database-management systems allow users to store and manage data in an easily accessible structured manner compared to simple file systems. Traditional file systems lack key advantageous characteristics provided by database-management systems. Database-management systems reduce data redundancy, as duplicate copies are no longer stored in multiple locations. Data security increases due to multiple levels of user access and a database-management system is capable of providing automated data backups instead of manual backups conducted by the end-user. Using a database-management system allows input constraints and increases data normalization while providing user concurrency. Data validation occurs before any changes are committed to the database and enforces atomicity within the system. Database-management systems support database-transaction concepts of atomicity, consistency, isolation, and durability (Haerder & Reuter, 1983).

The military currently uses several integrated database-management systems, which provide overarching support and increased military effectiveness. The Defense Finance and Accounting Services system provides standardized accounting and financial management for the Department of Defense. Marine OnLine is a personnel records database-management system for all military members in the Marine Corps Total Force System. Global Combat Support System-Marine Corps is the Marine Corps' logistical equipment database-management system. Each of these integrated systems can serve as a model for implementing proper training-record management.

A. ATTEMPTED SOLUTIONS

1. Marine-Sierra Hotel Aviation Readiness Program

According to the Department of the Navy (2012a), the Marine Corps currently has approximately 850 active duty and 250 reserve air-traffic-controllers, comprised of both enlisted and officers. There are 12 Marine Corps Air Stations, which provide ATC qualifications for the Marine Corps. Nine ATC detachments provide deployable ATC services to the Marine Air-Ground Task Force. Each facility has approximately 20 air-traffic-controller qualifications and the total qualifications differ for each facility based on number and type of flights controlled. The Navy also reports that qualification training time ranges from approximately 90 hours to 360 hours, which each qualification capable of exceeding the position by 200%. The air-traffic-controller detachments track qualifications used in a deployable environment. A deployable qualification typically predicates on obtaining an air station qualification.

The Marine Corps aviation community established a program to track deployable training and readiness event syllabus qualifications. The program, titled the Marine-Sierra Hotel Aviation Readiness Program (M-SHARP), is an online training database-management system formally implemented in August 2011 by the Department of the Navy. InnovaSystems International developed M-SHARP. The Department of Defense (2016) states that M-SHARP's intent is to track qualification fulfillments for Marine Corps aviators and ground-unit personnel that support the aviation community. The Department of Defense classifies air-traffic-controllers as ground support personnel. M-SHARP provides a central repository for storing, accessing, and modifying controller qualifications and designations. M-SHARP has additional capabilities tailored to aviators that are not applicable to air-traffic-controllers.

As shown in Table 1, training event records within M-SHARP store basic information for a specific training and readiness event (InnovaSystems International, 2016). The ATC's Tactical Training & Readiness Manual provides information such as the Syllabus and Description. InnovaSystems International (2016) states that individuals in instructor, supervisor, and administrator roles approve completed training-event records. Once these three billets have approved a record, it becomes part of the Marine's permanent record and individuals cannot change the record. If a Marine fails a training event, M-

SHARP permits an instructor, supervisor, or administrator to remove the record. Once a Marine earns a qualification, options exist to assign expiration and/or re-qualification timelines. M-SHARP marks these qualifications as expired, but they remain part of the Marine's record. M-SHARP also provides the capability for transferring Marine data between units when as Marines transfer.

Table 1. Training Information Stored within M-SHARP for a Training and Readiness Event. Source: InnovaSystems International (2016).

Heading	Remarks
Date	Date the training and readiness event is assigned. Click in the cell to display a calendar drop-down box and assign a start date.
Type	Click in the cell to display a drop-down box. Select the event type to assign an individual.
Personnel	Click in the cell to display a drop-down box of available unit members.
Syllabus	Click in the cell to select a Syllabus.
Description	Click in the cell to select from a drop-down box a list of Stages.
Start	Actual time the person started training for the Date assigned. Use 24-hour format (i.e., 0830).
End	Actual time the person ended training for the Date assigned. Use 24-hour format (i.e., 1630)
Total Training Time	Number of hours dedicated to training based on the START and END time for the date assigned. Only appears after the SAVE button is clicked.
Event Duration	The recommended time to complete the training for the training and readiness event. Only appears after the SAVE button is clicked.
Instructor	When checked, the training and readiness event is considered “Successfully completed” by the Instructor. When checked, the Supervisor check box is enabled.
Supervisor	When checked, the training and readiness event is considered “Successfully completed” by the Instructor and the Supervisor. When checked, the Admin check box is enabled.
Admin	When checked, the training and readiness event is considered “Successfully completed” by the Instructor, Supervisor, and the Admin. When the Admin box is checked and the SAVE button is clicked, the training and readiness event appears in the Crew Event Proficiency Report as a permanent record. Refer to *Section 4.2: Training & Readiness Report Submenu* for further explanation.

M-SHARP, although capable of tracking qualified personnel, cannot track a controller's detailed progression towards a specific qualification. As shown in Table 2, M-SHARP only has five possible status codes towards a training and readiness event: proficient, not proficient, no refly, incomplete, and complete (InnovaSystems International, 2016).

Table 2. Possible Qualification Levels within M-SHARP. Source: InnovaSystems International (2016).

Training & Readiness Event Status	Description
PROFICIENT	The crewmember has obtained the training and readiness code and the status is up to date.
NOT PROFICIENT	The individual has obtained the training and readiness code but it has expired (i.e., Expired).
No Refly	There is no refly interval associated to the training and readiness code/event.
INCOMPLETE	The individual has never obtained the training and readiness code but may need to acquire it as some future date.
COMPLETE	Not used in the M-SHARP program.

M-SHARP provides high-level reporting capability, but its lack of detailed performance metrics impedes analytical capabilities beneficial to air-traffic-controllers. Relevant examples are useful training analytics such as the correlation between a squadron's flight statistics and how it affects air-traffic-controller training timelines.

M-SHARP is a key component of the Marine Corps’ aviation training and readiness program. According to the Department of the Navy (2011), M-SHARP’s primary goal is “scheduling and logging Training & Readiness Events, comparing logged data to community readiness metrics, and formatting readiness data within the Aviation Training & Readiness Program Manual guidance” (p. 1-6). M-SHARP is used frequently to track deployable unit readiness. The ATC Tactical Training & Readiness Manual mandates that detachments use M-SHARP to calculate the core model minimum requirement, which is the basis for evaluating unit readiness. Units who do not meet the minimum requirement are required to focus their training and meet the minimum requirement within the time constraints prior to deployment (Department of the Navy, 2016).

Although M-SHARP is efficient in tracking training and readiness events, M-SHARP focuses solely on deployable ATC qualifications. Some of these qualifications are similar, but M-SHARP's qualifications are Boolean in nature. M-SHARP lacks the capability of tracking individual controller progression, which is essential for use at air stations. As a result, air stations had to develop local training database-management systems to track the detailed controller progression data not tracked by M-SHARP.

2. Local Databases

For the task of manually tracking detailed controller training statistics, units developed individual training database-management systems. These systems use software such as Microsoft Access, Microsoft Excel, or MS-DOS programs. These developed database-management systems are client-based, typically requiring use of a single computer. Training database-management systems can be placed on a network; however, Pavlo et al. (2009) states that performance drops significantly with large data transfers and with computations sent remotely to the data instead of the reverse. Only one user can currently modify such a training database-management system at a time. As a result, the unit's training manager must devote at least an hour daily to update the database-management system with the previous day's training data. This task increases substantially after a weekend or holiday period, as controller training occurs daily.

These training database-management systems, while capable of tracking and processing controller training data, have significantly expanded in size over the years. As a result, the systems require significant start-up times and are prone to errors and crashing, resulting in possible data corruption and data loss. The systems also lack backup capability and version control, resulting in users regularly copying the entire training database-management system onto shared folders in case of local computer corruption. Blischak, Davenport, and Wilson (2016) state that without version control, users lack teamwork capability, safe workflows, and the ability to modify the system without the risk of corruption. Administrators create multiple backups of the existing systems prior to system modification to ensure the existing data is not corrupted.

B. SIMILAR APPLICATIONS

Lee et al. (2015) identified a hospital, which needed to migrate an existing medical record system storing oncology patient data to an updated database-management system. The hospital's existing record system used Microsoft Excel to store patient notes and the system users manually entered the patient data. System users then manually transferred the data to a special form, requiring the user to analyze every text field and enter the corresponding value into an adjacent field. Once this step was complete, the users exported the data to be analyzed using statistical analysis software.

The hospital's existing record system had storage limitations and questionable data quality. Lee et al. (2015) found that “data input and analysis without a database run a higher risk of incorrect data entry, patient exclusion, and a higher risk of introducing duplicates” (p. 2). Developers initially proposed adopting an existing clinical research database, however, Meineke et al. (as cited in Lee et al., 2015) found that re-tasking a separately designed database-management system would not be specific enough for the oncology research and would need additional development for all the required calculations.

According to Lee et al. (2015), the hospital's key focus areas for improvement involved developing a front-end graphical user interface for user simplicity, automated calculations, and creating statistical outputs from the entered data for easy data examination. As part of the system's development, the hospital was required to develop a database-management system in compliance with the Health Insurance Portability and Accountability Act. This required establishing strict policies regarding data access, storage, and access audits.

Lee et al. (2015) found that the hospital greatly increased work speed after implementing the new database-management system. Customized reports were easy to access and allowed for multiple users concurrently interacting with the database-management system. The hospital did identify, however, that significant training time was required to teach the new front-end graphical user interface and for incremental database testing prior to implementation. A database administrator was necessary to maintain the database-management system, which performed events such as backups, interface modifications, and software updates.

After analyzing this case study, we identified key requirements for implementing a successful database-management system. Before the hospital developed their database-management system, they reviewed the current electronic medical record system in use and identified key requirements for successful system implementation. Creating a new system without identifying key requirements can lead to functionality loss, as stated in the first chapter. The hospital also identified possible methods for streamlining workflow processes. This eliminated redundant processes within their current electronic medical record system and introduced faster and more detailed data analysis.

THIS PAGE INTENTIONALLY LEFT BLANK

III. APPLICATION DESCRIPTION

This chapter identifies specific problems in current training-records practice that the design and program we developed will address. We describe issues encountered at individual units and explain how units are currently mitigating these issues. We describe assumptions made about the program, its intended audience, and the applicable orders and regulations.

A. THE CURRENT TRAINING-RECORD SITUATION

Units are currently using training database-management systems developed over several years. Due to the differences in local flight operations and the technical skill of individuals attached to a unit, multiple individual systems were developed. Smaller units could successfully track training qualifications using spreadsheet applications such as Microsoft Excel, while other larger units used larger applications such as Microsoft Access or MySQL. These systems have outdated platforms, and individuals currently maintaining these systems may not fully understand the system developer's intended processes and workflows. Unless a government civilian still stationed at that unit developed the training database-management system, there is a high likelihood of multiple users involved in the system's maintenance lifecycle instead of by the original developer. Since many training database-management systems have been independently developed and operated, significant issues arise when an individual transfers between units. Electronic records are typically not transferrable, as units may use different platforms and lack a centralized data migration process.

The Chief of Naval Operations (2002) states that the qualifications, operations, and individual controller status must be tracked in accordance with Naval Air Systems Command guidelines. The Marine Corps billets a Marine Training Chief to fulfill this requirement. Part of the Training Chief’s duties include establishing a local air-traffic-controller training program, conducting classroom training, developing tests to evaluate the results of the scheduled training, and maintaining the unit's ATC certification and qualification records. The Department of Defense (2004) states that a Marine staffs the Training Chief billet unless the ATC Training & Readiness Office authorizes a temporary waiver. This is a mandate to improve record keeping and promote Marine involvement within the development process.

1. Workload Duplication

When the Marine Corps select a Marine for the Air-Traffic-Controller Military Occupational Specialty (MOS), the Marine attends the ATC basic course (ACA1). The ACA1 issues a MACCS Performance Record (MPR) to the Marine upon graduation from the ACA1. The MPR is a hard-copy folder designed to provide the Marine's ATC history in a standardized professional format.

The MPR has four sections; administrative information, MOS training qualifications, managed on-the-job training and general training information. The administrative information section documents MPR audit records and all formal training history, which includes ATC schools and MOS progression. The MOS training qualification section documents MOS qualifications and designations. Documented contained within the on-the-job training section are academic training records, training time summaries, and training and readiness syllabus event evaluations. Individual deployment records, professional military education, and other ATC miscellaneous information not previously listed are stored within the general training information.

Initially, the MPR only contains the student's ACA1 school training record and graduation certificate. The Chief of Naval Operations (2002) states that the MPR must be maintained at either the ATC facility or the squadron's S-3 Operations section based on where the Marine is assigned after the ACA1. The Marine’s MPR is maintained by the Training Chief while the Marine is assigned to the unit. When a Marine transfers between units, the Marine should make copies of their MPR in case the MPR is lost or damaged in transit.

When a Marine joins a new unit, the training office's first requirement is to enter the joining Marine's information into the local unit's training database-management system. It is up to each individual unit to determine which records of previous qualifications are stored within their training database-management system. The training office enters basic personnel information such as full name, EDIPI, age, sex, and Military Occupational Specialty (MOS) qualifications in the unit's system for later reference.

The Marine must complete the initial data entry process upon every unit transfer. The only exception to this policy would be if the unit had the Marine's information already due to the Marine stationed at that unit previously. Every unit electronically stores a Marine's data differently using their individual systems. The hard copy MPR and M-SHARP are the primary means of long-term global data storage, with local unit training database-management systems being a secondary backup holding more detailed training metrics.

M-SHARP is capable of transferring Marines between units but lacks key data such as detailed personnel information and qualification progress performance. Local training database-management systems must store this data and these systems do not communicate with M-SHARP. As a result, only basic qualification data electronically transfers between units within M-SHARP.

2. Qualification Timeline Training

a. Government Civilians

Government civilians are required to earn and maintain currency on all possible qualifications within the air station. Typically, units hire civilians based on previous qualification records and train civilians at a priority level. Doing so allows a civilian to be fully facility-rated and provide necessary facility manning and instructor training to Marines and other civilians. Civilians remain permanently stationed at a unit unless they retire, choose to relocate, or are terminated for legal or performance issues.

b. Marine Officers

The Marine Corps assigns ATC Officers the MOS 7220, Air-Traffic-Controller-Officer, after graduating the ACA1. Officers are required to earn the tower ground qualification and radar final controller qualification. After obtaining these qualifications, the Marine Corps typically assigns ATC officers to deployable units and the officers serve in these billets for their duration as a 7220 MOS. Officers change to a 7202 MOS after promotion to the rank of Major. At this time, the ATC officer performs in more generalized MACCS billets and not solely within ATC billets. During their time as a 7220 MOS, officers follow a structured career progression model established within the Marine ATC Tactical User Training & Readiness Manual. These skills focus on mission proficiency and the Marine Air-Ground Task Force, not individual controller qualifications.

c. Marine Enlisted

Once enlisted Marines graduate from the ACA1, the Marine Corps assigns the enlisted Marines the MOS 7251, Air-Traffic-Controller-Trainee. Within nine months of graduating the ACA1, Marines are required to earn the qualifications for MOS 7257, Air-Traffic-Controller. The MOS 7257 is a prerequisite for all subsequent enlisted MOS qualifications. As enlisted Marines progress in their career, they are required to earn three major qualifications prior to being eligible for promotion to the rank of Master Sergeant and assigned the MOS 7291, Senior Air-Traffic-Controller. The required major MOS qualifications are MOS 7252 - Air-Traffic-Controller-Tower, MOS 7253 - Air-Traffic-Controller-Radar Arrival/Departure Controller, and MOS 7254 - Air-Traffic-Controller-Radar Approach Controller. Marines must earn their initial major MOS within three years of graduation from the ACA1 or they will be revocated out of the ATC MOS. The Marine Corps recommends ATC Marines to obtain a subsequent qualification within six years of graduation from the ACA1 and must earn a subsequent qualification within nine years of graduation from the ACA1. The Marine Corps requires the final major qualification within 12 years of graduation from the ACA1. The Marine's unit administrative office must place a copy of each qualification record within the Marine's Official Military Personnel File. Headquarters Marine Corps electronically maintains the Official Military Personnel File and the file is accessible through the unit's administrative office or via Marine OnLine. The Official Military Personnel File does not store detailed training records and only stores the record necessary to add an additional MOS to a Marine's military record within Marine OnLine.

The Marine ATC training pipeline is extremely complex due to multiple agency involvement and detailed training requirements. The Marine ATC Tactical Training and Readiness Manual states:

> Per the NAVAIR 00-80T-114 Air-Traffic-Control NATOPS Manual, CNO N885F is the ATC facility classification authority for the Department of the Navy (DON). ATC facility classification determines which ATC skill sets DON air-traffic-controllers can train to and qualify for. In accordance with the NAVAIR 00-80T-114, MARADMIN 229/04 lists ATC services provided, ATC skill sets trained to, MOS...ratings available at MCAS and MCAF ATC facilities based on CNO N885F assigned ATC facility classifications. The policy set forth in MARADMIN 229/04 remains in effect until cancelled by Headquarters Marine Corps/Aviation. (Department of the Navy, 2012a, p. 2-10)

The Chief of Naval Operations (2002) states that with Marines being assigned collateral duties, deployments, and billets outside of the ATC field, qualification progression does not always fit the required timeline as prescribed by Naval Air Systems Command. If a Marine is unable to meet the required Naval Air Systems Command qualification timeline, the Marine's current unit is required to request a waiver to Headquarters Marine Corps/Aviation Expeditionary Enablers Branch-25 to stay within the ATC MOS. Within that waiver request, the unit must submit a detailed training plan for the Marine. If approved, the waiver approval is stored within the Marine's MPR and the Marine has a specified timeframe to earn the MOS qualification. A Marine's unit is expected to expedite the Marine's training if the Marine is on an approved waiver or is approaching a major MOS qualification deadline. This is done to ensure the Marine is not revocated out of the MOS.

Typically, a reduced qualification timeline is identified only when the Marine arrives at the new unit and the Marine's MPR is reviewed. If the gaining unit were proactive, they would contact the releasing unit to see if the Marine is on an existing waiver or if the units projects that the Marine will need a waiver in the near future. The Department of the Navy (2013) states that projecting qualifications is beneficial to the new unit, as the training pipelines for the three major MOS qualifications are typically forecast 18–24 months in advance due to both a backlog of controller training and supporting qualifications. If a new unit receives a Marine who already has a major MOS qualification, the unit can train the Marine in half of the required training time for that position and use the Marine to support their staffing goals.

3. Report Generation

An MPR stores a Marine's air-traffic-control training data. Data within the MPR includes designation certifications, MOS qualification records, deployment records, and academic-training records. Units are required by Naval Air Systems Command to maintain a Marine's MPR while the Marine is within their unit, and must forward the hard-copy MPR to the next unit after the Marine has been re-assigned. The Chief of Naval Operations (2002) prescribes that a unit is additionally required to keep the hard-copy MPR for a period of six months if a Marine is revocated from the air-traffic-controller MOS, separates from the Marine Corps, transfers to the Marine Corps Reserves, or retires while assigned to their unit. A unit must also retain historical airfield data as prescribed by the Naval Air Systems Command. Units must retain daily airfield records for six months, while broader encompassing data such as airfield modifications, airport usage, and airfield correspondence require retaining historical data for at least six years.

Marines use a single hard-copy MPR throughout their career, and inevitably, there will be circumstances when the MPR becomes lost, damaged, or destroyed. If an MPR is not repairable, the Marine is required to re-create their MPR. Units advise Marines to make photocopies of their MPR prior to transferring duty stations, but this is usually not completed. Current units can only replicate some of the Marine's MPR records. Marines must obtain other records within the MPR by contacting each of their previous units if the records were not entered into the current unit's training database-management system. Marines can retrieve basic MOS qualification records from their electronic Official Military Personnel File maintained by Headquarters Marine Corps, but detailed training data such as individual facility position qualification metrics will be lost.

Based on a unit's record-keeping policies aside from those required by Naval Air Systems Command, a Marine may only recover limited historical information. The Marine's present unit will typically only be able to re-create the detailed training data on positions qualified while at that current unit. This capability additionally depends on what type of data is being stored within the unit's training database-management system. A Marine can contact their previous commands for any training records, but the unit may not have retained them. The Department of the Navy (2011) currently only dictates that the “Aviation Ground communities shall use Performance Records (PR) as prescribed by the individual communities in coordination with the syllabus sponsor” (p. 2-36) and does not specify what additional electronic methods of storage must be used.

As stated by InnovaSystems International (2016) publications with guidance from the Department of Defense, individual units are required to submit standardized monthly, quarterly, and yearly training reports to their regional units. The training reports detail current trainee information such as current assigned crew, training start date, projected qualification date, and any additional remarks from the crew or facility. If the controller were currently training on a position, the unit would enter total hours trained and overall training total percentage on the report (e.g., 10/5% means ten hours trained, 5% overall complete).

As the trainee data updates daily, the units must calculate these reports by tracking controller progression within the local unit database-management systems. When a report is due to be submitted, the unit creates the reports by manually transferring data from the local training database-management systems to a provided Microsoft Word or Adobe Acrobat PDF template, which the unit then emails to their regional headquarters. These reports allow regional headquarters to view point-in-time snapshots of how the individual unit's current training programs are progressing.

The Marine Corps can use the reported data to track and develop aggregate performance metrics at a regional command level. An example could be a regional command identifying that Precision Approaches to Radar training are lower than average for certain facilities within their regional command. The regional commander will be capable of using this data as a talking point to the Marine Aircraft Wing to support increasing Precision Approaches to Radar to regional facilities. Precision Approaches to Radar are a key component of air-traffic-controller training; not only do they allow aircraft retrieval in times of greatly reduced visibility, but they also develop the supporting qualifications needed for the MOS 7253, Air-Traffic-Controller-Radar Arrival/Departure Controller, and MOS 7254, Air-Traffic-Controller-Radar Approach Controller.

4. Data Validation

To effectively manage controller training, the Training Chief maintains a local training database-management system of individual obtained position qualifications, training progression towards a position qualification, facility designations, and a position's current proficiency. Typically, the Training Chief will inherit the unit's existing system when the Training Chief transfers to that unit. Data stored within the unit's training database-management system can potentially have over a decade's worth of historical records, as this information can possibly allow more accurate projected qualification timelines since every unit has different training environments.

As there is a high likelihood that the incoming Training Chief is inheriting a new training database-management system at their assigned unit, the Training Chiefs are required to simultaneously learn this system while also obtaining all facility qualifications at their assigned ATC facility, another requirement of the Training Chief billet. If the Training Chief encounters a records-management issue with the training database-management system after the Training Chief completes handover from their predecessor, the Training Chief must either track down the previous chief who has since transferred to another unit or identify another individual within the unit who is familiar with the system. The issues usually encountered by a Training Chief relate not to software corruption issues, but instead to procedure workflows or mitigation strategies due to current software limitations.

As the Training Chiefs transfer to different units, institutional memory on how to manage specific software records-management issues is often lost due to poor program documentation. With the training database-management system being a one-off design, insufficient documentation can result in the improper use or neglect of key features, quirks, or required system updates. The Training Chief typically establishes themselves in one office, with resident files such as the examined training database-management system being stored on their individual computer. Usually included within these personal files is documentation for the developed system. Unfortunately, hardware quality and network capability are not at a premium, and individuals must either store all their critical files on a shared network drive or external device. If the local computer fails and the unit did not properly back up the documentation files, there is no available technical support or other means of understanding the developed system.

Training database-management systems can eventually become either defunct or a mere shell of their initially designed capabilities over time.

Customized free, or “homegrown,” software provides immediate benefits over a commercial off-the-shelf system purchased for a specific purpose. Developers can adopt customized free software to best serve an individual unit and can reduce commercial software acquisition and licensing costs. Downfalls of using customized in-house software, however, become evident over time. Dryden (1998) identifies the issue with homegrown software being “everyone has custom tools and scripts, but the problem is that they are all developed by gurus, and gurus leave the company...they can sling together something of beauty, yet almost never document their code to any level of professional standards” (p. 49).

A unit's training database-management system stays only as strong as the member currently maintaining the system, typically as a result of improper documentation coupled with non-standard systems due to lack of formal software development methods. If a unit is fortunate enough to have a Training Chief who is knowledgeable of the system, whether due to being previously stationed at the unit or having a software-related background, the Training Chief could upgrade the system to reflect current requirements. These training system upgrades are usually minor in detail, but have a high risk of corrupting the current database. An example of this is the migration from social security numbers to EDIPI as a primary means of identifying individuals within the database.

With training data being stored locally at a unit, there may be a possibility of data invalidation due to improper record keeping. With local systems, there is no higher level of oversight from an external source to validate proper record storage. Individuals can improperly enter data and units may not properly update newly published training guidelines to their training database-management system as intended by higher headquarters, resulting in problems such as incorrectly calculated training times. Marine Aviation Weapons and Tactics Squadron One, directed by the Department of the Navy, provides standardized training, guidelines, and support to the ATC Training & Readiness Office (2012). The ATC Training & Readiness Office publishes these guidelines in the Marine ATC Tactical User Training & Readiness Manual.

Siha and Saad (2008) identify that data validation is crucial for training database-management systems. Inspection teams conduct yearly audits, but systems currently in use lack continuous data validation. Improper data restrictions may limit operator input, which results in users developing individual incorrect methods to bypass restrictions. A controller having more qualifications than allowed due to improper database design restrictions is an example of this restriction. The observed method within the sample database to bypass the qualification restriction was adding a new member to the database with the notation "-1" added to the end of the EDIPI. This clearly indicated to a user that the two members were the same individual, but removed any analytical capability between the two records.

When the ATC Training & Readiness Office issues directives that change training data methods of reporting, training database-management systems can become ineffective immediately. To update the system's procedures and workflows, changes may require minor updates such as formatting or major updates such as using an EDIPI as a primary identifier instead of a social security number. A local user can accomplish minor updates, but major updates may require someone deeply knowledgeable of the current training database-management system since multiple workflows can be dependent on record identifiers.

If system maintainers do not properly update the training database-management system, there runs a high risk of incorrect record calculations occurring. Poorly detailed tracking can ultimately result in either an excess of students awaiting training or a gap at facilities for training personnel, as qualification progression data is used at the regional and headquarters-command levels to forecast training pipelines. These training pipelines can also affect Marines further along in their career path.

B. ASSUMPTIONS MADE FOR A NEW SYSTEM

In designing a new system, we assume that Training Chiefs at the unit level and individuals managing ATC data at the regional and higher levels will be the users. The training database-management system would focus on integrating data from multiple units for higher-level data analysis. By using one training database-management system for the entire air-traffic-controller fleet, a unit can eliminate a large amount of manual data entry regarding Marine’s personal and historical training data. As stated previously, M-SHARP does not track detailed performance and personal data and this information must be stored in local systems. The initial data entry for a Marine graduating the ACA1 will still be required, but a unit's training office will no longer have to enter the basic personnel information when a new Marine arrives, as only the Marines coming direct from the ACA1 would not be in the developed training database-management system. A unit can greatly simplify the process of recreating a hard-copy MPR, as all training records for their career will be within the developed electronic training system.

With individual units submitting training reports on a pre-planned schedule, units provide regional commands an overview at specific intervals. Regional commands can currently request reports on an as-needed basis from local units, but the developed system can provide regional commands with the capability to analyze training data for a specific period. Having a centrally located data repository can eliminate the requirement for detailed manual report generation. With all of the pertinent data capable of pre-population from the developed training database-management system, local units could significantly reduce the amount of time required to manually calculate training totals and can reduce data entry mistakes.

The developed training system must adhere to the Marine Corps Cybersecurity Program and the Department of Defense Operations Security Program Manual for PII compliance. Having the training system follow the Marine Corps Cybersecurity Program and Operations Security Program Manual would comply with all Department of Defense, Federal, Department of the Navy, and Marine Corps requirements to ensure PII violations do not occur (Department of Defense, 2008; Department of the Navy, 2012b).

THIS PAGE INTENTIONALLY LEFT BLANK

IV. PROGRAM DESCRIPTION

A. HARDWARE/SOFTWARE ENVIRONMENT SPECIFICATIONS

The system we developed as a prototype operated on a Dell Latitude E6500 laptop. The Latitude E6500 laptop ran Windows 7 Enterprise, Service Pack 1. The Latitude E6500 laptop had a 2.4-gigahertz P8600 Intel Core 2 Duo Processor and four gigabytes of RAM.

The developed training database-management system used SQL, Visual Basic for Applications, Bash, and SQL-Loader programming languages. The Dell laptop used Microsoft Access 2013 and Microsoft Excel 2013 to analyze the sample database. We established a development environment in a virtual environment obtained from Oracle. The minimum operating requirements for the virtual machine were one gigabyte of RAM, 15 gigabytes of hard drive space, and a 2.0-gigahertz processor. The recommended operating requirements are two gigabytes of RAM and 15 gigabytes of contiguous hard drive space (Oracle, 2016). We managed the virtual machine using Oracle VM VirtualBox Manager software, version 5.0.22.

Oracle's pre-built virtual machine operated with Oracle Linux 7, Oracle Database 12c Release 1 Enterprise Edition, Oracle SQL Developer, and Oracle's Application Express. Oracle Database 12c was version 12.2.0.20.96, IDE version 12.2.1.0.42.151001.0541. SQL Developer was version 4.1.3.20.96, which used Java platform 1.8.0_65.

B. SAMPLE DATABASE ANALYSIS

Our sample unit, located at MCAS Beaufort, developed a training database-management system using Microsoft Access 2003 and stored the data within a Microsoft Access database file. The database file contained 350,389 records. During initial analysis, we identified that the database had four missing Data Access Object dynamic-link library dependencies referenced from Microsoft Access 1997. These missing dynamic-link library files resulted in Microsoft

Access unexpectedly quitting while analyzing the database. The database continued the fatal errors until all dependent plugins were disabled.

The sample training database-management system had password protection using the built-in Microsoft Access protection features. After logging in with the provided credentials, the sample database presented a user with a front-end graphical user interface designed using Microsoft Access forms to manage the system. The database consisted of 50 tables. When we reviewed the database tables, we identified 12 tables that were empty and did not hold any data. The empty system tables were titled "Archive $NAME," where $NAME was an already existing table with data stored within it. The empty database tables mirrored the design of the copied tables.

We believe a user improperly created these tables when attempting to back up existing records. There were additional tables within the database titled "Archive_$NAME" or "$NAME Archive," however, these actually had archived data relating to the table. Five of the archived tables only held data for one year and the year differed in every table.

Six tables within the database contained Marine-specific information not related to air-traffic-controller training. Examples of the stored data include rifle and pistol scores, completed Marine Corps Institute courses, meal and rifle card numbers, and personal information such as height, weight, and contact information. All of the Marine-specific data within the sample database is currently stored within Marine OnLine, but may not have been at the time the developers created the training database-management system.

The developers divided the tables storing relevant records into facility logs, position logs, flight operations, and MACCS record jacket data. The developers largely grouped the tables by date and then by relevant key data such as EDIPI or controller-operating initials. The developers generically labeled the fields within the tables; one table had 141 fields, named "remark1" through "remark47," "equipment1" through "equipment47," and "UTC1" through "UTC47."

Each grouping (remark, equipment, and UTC) corresponded as a single log entry. The developers used controller-operating initials within the tables to identify those who were on position at the time of the log entry.

Once we analyzed the tables within the database, we found that the database had inconsistent primary keys for identifying unique records. The inconsistent primary keys appeared to be a result of the multiple database updates and modifications over the years, presumably by different individuals that were attached to the unit at the time. As a result, certain fields used as primary keys were blank before a specific date, whereas other fields lacked activity after a certain date. Example primary keys included full name, controller-operating initials, date, or EDIPI. Some fields concatenated other primary keys and used the result as a primary key. Out of the identified primary keys, EDIPI was the only true indicator that could guarantee unique records. Multiple individuals exist within the Marine Corps that share names, and controller-operating initials could be re-used between controllers, just not concurrently. Date-record formatting was also not consistent within the Microsoft Access database. Out of the 50 tables within the database, only one table did not have blank values within the table records. Blank values within a record result in database ambiguities. Blank values could mean unknown values, not applicable values, or the values were in other tables or records.

Within the training position qualification table, the developer stored the records in the format EDIPI, name, facility, date, and position as shown in Table 3. The developer used the EDIPI and name once, however, the facility, date, and position had up to 27 qualifications stored within one record. Due to both the EDIPI and names in the table, there were records with an invalid EDIPI but correct name so the system accepted the record entry. After the initial invalid EDIPI and name were entered, additional qualifications were added to invalid EDIPI due to the database displaying a username taken from the table instead of looking up a name using an EDIPI. The EDIPI remained invalid in the record, as the front-end user only viewed the controller's name.

Table 3. Sample Fields within Training Position Qualification Table

EDIPI
name
current_facility
start_date
qual_date
position
facility
1start_date
1qual_date
1position
1facility
2start_date
2qual_date
2position
2facility

C. SYSTEM DEVELOPMENT

A primary goal was data integrity in the training database-management system to the greatest extent possible. When reviewing the sample database, we identified records that the system was not processing due to improper user-entered values. This resulted in the system not properly joining the records and filters improperly excluding relevant data. Limiting the amount of end-user entries and implementing strict data implementation would reduce many data errors within the system. Some records even lacked enough identifying data to prevent us from manually associating those records with individuals.

In the sample training database-management system, personal data, controller qualifications, and current duty-station information were all stored in the same table. Despite making it easier to view all the stored data via the back-end interface, having a single table prevents installing different user-permission layers. Additionally, hard-coding many fields may cause problems in the future if additional fields are that would exceed the amount of provided space. We provided an example of this in Chapter III, Section 4.

We created four key entities for managing the training records: personnel data, controller qualifications, controller logs, and training logs. This delineated data boundaries and helped identify key relationships between the established entities. By identifying key data attributes and properly assigning field data types, we established clear table relations and ensured unique constraints within the system. We assigned primary and several foreign keys to enforce data integrity. We designed the system for maximum database normalization. Lack of database normalization was a key issue in the sample system, resulting in significant data redundancy and poor data integrity.

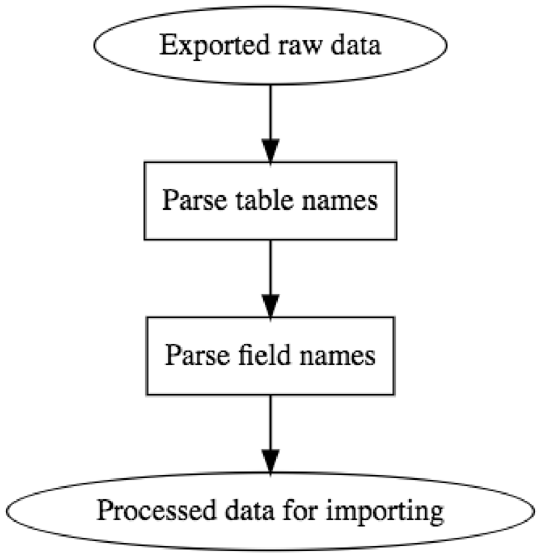
When developing a relational database, data should be stored in first-normal form. First-normal form requires a primary key, no repeating groups of data, and every column to be unique within the table. An example of a first-normal form violation is one record having the fields: EDIPI, Qualification1, Qualification2, and Qualification3. Instead, each qualification should be stored as a separate record (e.g., three records of EDIPI, Qualification). With the original record, a controller is limited to three qualifications. If a controller has less than three qualifications, the database must allow NULL entries and can lead to insertion anomalies. Storing data in this method also requires a developer to accurately predict the correct number of qualifications a controller will ever have. With a first-normal form design, a controller can have an unlimited amount of qualifications within the system.

In the sample system table described in Chapter IV, Section 2 that limited the number of daily logs to 47 entries, the qualification records were also limited to only 17 possible qualifications per user due to the original developer's improper design restrictions. Although the sample unit had less than 17 qualifications at their facility, controllers had additional qualifications from previous units. Once a controller exceeded 17 qualifications, the Training Chief had to create an entirely new record within the database to store additional qualifications. Storing data in first-normal form would eliminate both of these situations.

Due to multiple billets within the unit, we established three permission levels within the developed training database-management system. With permissions ranging from read-only access for unit administrative positions, providing back-end users enforces the principle of least privilege. Instituting principle of least privilege allowed us to limit a user's access to key data within the system. We also needed to establish data concurrency and data consistency within the designed system. Oracle's Application Express, used to develop the front-end graphical user interface, enforces data concurrency via Automatic Row Processing.

D. SAMPLE DATA EXPORTATION

We developed a Visual Basic for Applications script that would export every table within the sample database to individual comma-separated-value files. By creating individual files, we enabled data portability and we were able to properly format the raw data without altering the original database. As diagrammed in Figure 1, we divided the single script into two separate scripts and completed the file processing in batches, as initial attempts to export the entire database within one script resulted in Microsoft Access terminating unexpectedly.



Exported raw data comes from Access database file. Processed data is stored in comma-separated-value files.

Figure 1. Processing Exported Data

We executed both scripts in Microsoft Access. With units managing databases established over 10 years ago, simplicity was a priority when designing the migration tools. We created Individual Visual Basic for Applications modules to reduce code complexity and allow maximum backwards-compatibility. We did not design graphical user interfaces as part of the export script, as we designed this code for one-time use to transfer existing data. Once units fully transition over to the developed training database-management system, the units would no longer use the existing Microsoft Access database file. The data in the current training database-management system would be left in case of any required historical reference.

The first Visual Basic for Applications script processed each Microsoft Access database file table name, removing any characters that would be illegal in an Oracle SQL database, and creating a comma-separated-value file for each table. Oracle states that table and field names have the following restrictions:

1. Names must be from 1 to 30 bytes long.
2. Non-quoted identifiers cannot be Oracle Database reserved words. Quoted identifiers can be reserved words, although this is not recommended.
3. Non-quoted identifiers can contain only alphanumeric characters from the database character set, the underscore (_), dollar sign ($), and pound sign (#). Database links can also contain periods (.) and "at" signs (@). Oracle strongly discourages you from using $ and # in non-quoted identifiers.
4. Non-quoted identifiers must begin with an alphabetic character from the database character set. (Oracle, 2017, p. 19)

As the Visual Basic for Applications script created each table, the script then exported every table's data to an individual comma-separated-value file. The Visual Basic for Applications script verified that an existing file with a matching filename did not exist before creating the comma-separated-value file. If a file existed, the script displayed an error message notifying the user that continuing would overwrite an existing file, and the script terminated. The Visual Basic for Applications script did not export any of the Microsoft Access database file system tables. When code execution was complete, the script displayed the total number of processed tables and allowed us to validate that the correct number of tables were processed. Having filenames matched to table names provided an additional level of validation.

We developed a second Visual Basic for Applications script that processed the individual exported comma-separated-value files after exportation. The second script focused on parsing all field names within every table. The script identified any field names reserved within Oracle, identified in Table 4, or those that contained illegal characters.

Table 4. Oracle Reserved Words. Source: Portfolio et al. (1996, p. B-2).

ACCESS	ELSE	MODIFY	START
ADD	EXCLUSIVE	NOAUDIT	SELECT
ALL	EXISTS	NOCOMPRESS	SESSION
ALTER	FILE	NOT	SET
AND	FLOAT	NOTFOUND	SHARE
ANY	FOR	NOWAIT	SIZE
ARRAYLEN	FROM	NULL	SMALLINT
AS	GRANT	NUMBER	SQLBUF
ASC	GROUP	OF	SUCCESSFUL
AUDIT	HAVING	OFFLINE	SYNONYM
BETWEEN	IDENTIFIED	ON	SYSDATE
BY	IMMEDIATE	ONLINE	TABLE
CHAR	IN	OPTION	THEN
CHECK	INCREMENT	OR	TO
CLUSTER	INDEX	ORDER	TRIGGER
COLUMN	INITIAL	PCTFREE	UID
COMMENT	INSERT	PRIOR	UNION
COMPRESS	INTEGER	PRIVILEGES	UNIQUE
CONNECT	INTERSECT	PUBLIC	UPDATE
CREATE	INTO	RAW	USER
CURRENT	IS	RENAME	VALIDATE
DATE	LEVEL	RESOURCE	VALUES
DECIMAL	LIKE	REVOKE	VARCHAR
DEFAULT	LOCK	ROW	VARCHAR2
DELETE	LONG	ROWID	VIEW
DESC	MAXEXTENTS	ROWLABEL	WHENEVER
DISTINCT	MINUS	ROWNUM	WHERE
DROP	MODE	ROWS	WITH

The Visual Basic for Applications script included enforcing a maximum character field length, eliminating spaces, and having fields begins with alphanumeric characters. The script additionally renamed all fields to lowercase, as field names are not case-sensitive within Oracle and we wanted to eliminate reference issues later in development.

E. DATA IMPORTATION

After we generated the comma-separated-value files, we manually transferred the files to the Oracle virtual machine. In future use, one could establish a network connection for file transfer or provide additional scripts to remotely connect to the Microsoft Access database files. We can run these scripts at off-peak hours to have minimal workload disruption.

Once we transferred the files to the Oracle virtual machine, we used SQL*Loader to import the comma-separated-value files. We created separate SQL*Loader scripts for every file, which allowed us to individually view any generated error messages when executing SQL*Loader. Each SQL*Loader script required three components: a control file, a batch file, and the data to be imported. The batch file is an executable file which runs the SQL*Loader commands. The batch file specifies destination database information and the SQL*Loader output files. The control file specifies:

1. Where SQL*Loader will find the data to load.
2. How SQL*Loader expects that data to be formatted.
3. How SQL*Loader is configured (memory management, rejecting records, interrupted load handling, and so on).
4. How SQL*Loader manipulates the data being loaded. (Oracle, 2002)

We created a separate schema to store the values from the comma-separated-value files. By storing this raw data separately, we could effectively use this as a staging schema. The naming structure for the created tables and field-names within the tables closely matched those within the sample Microsoft Access database. We appended the text "_RAW" to every table name, as an additional indicator to users. By mirroring the naming within the Microsoft Access database, we provided users with an easier method to trace issues while importing data.

We imported each comma-separated-value file into its separate table using SQL*Loader. After we tested the individual SQL*Loader processes, we created a single script that would execute all the SQL*Loader processes. A single script increases automation capability and can simplify batch processing during off-peak hours.

F. DATA PROCESSING

After the data was successfully imported into the staging schema, we first created a user within the developed production schema with read-only access to the staging schema. This prevented accidental data modification. If a database user within the production schema needed to modify the staging schema, they could still have access internally within the staging schema.

After we enabled read-only access to the staging schema, we identified key data for importing into the production schema. Multiple tables and fields within the staging schema involved non-ATC data and were not relevant to the production schema. We focused on correctly processing users within the staging schema according to their EDIPI, which served as the unique identifier within the production schema. We developed queries to validate our production schema, ensuring we enforced all unique constraints and identified any invalid raw data. Once we confirmed our developed queries, we processed the data into our production schema.

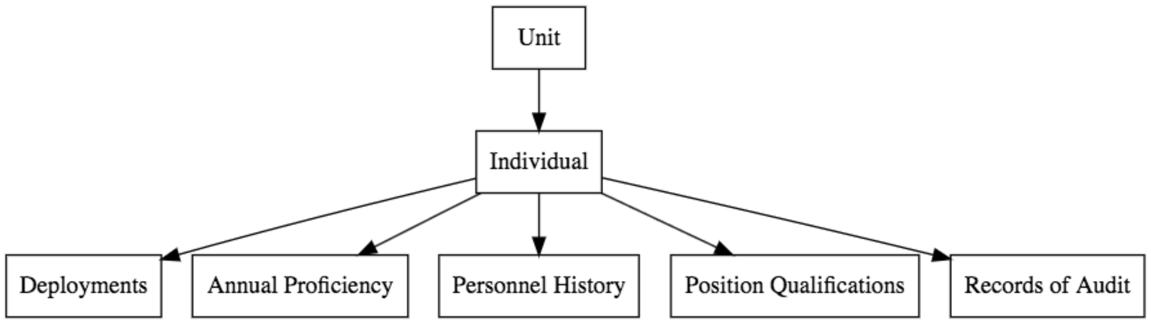
G. END-USER REPORTS

We developed a front-end graphical user interface to our production schema using Oracle's Application Express. The interface presents users with an initial login screen. Users require initial registration and system access with the developed system maintainers, who grant permission levels based on the unit's request. This provides an additional level of credential verification and currently mirrors user registrations used with systems such as M-SHARP.

User permissions would allow either view-only or read and write access to the stored records. Members in key billets such as Officer-in-Charge, Staff-Non-Commissioned-Officer-in-Charge, Training and Standardization Officer, Radar or Tower Chief, and the Training Chief would be granted higher permission levels as needed, and lower levels of leadership such as Crew Chiefs would be granted view-only access. Enabling view access to Crew Chiefs provides them with the ability to track training qualifications without needing to contact a key billet member while still enforcing the principle of least privilege.

After logging in, the system presents members with the capability to generate MPR records, or add, modify, or update existing data, in addition to generating MPR records. Unit members would be restricted to members within their local facility, whereas members at higher regional levels would have access to multiple facilities within their purview. Figure 2 shows the workflow for viewing a Marine's MPR. When an end-user logs into the system and selects the option to generate MPR records, the system enables them to select a unit for report generation. The system prepopulates this field with the end-user's currently assigned unit and marks the field read-only if an end-user does not belong to a regional level unit.

The system then enables the end-user to select an individual currently assigned to the selected unit and view the records currently stored within the MPR. The front-end graphical user interface displays the selected individual's record in the web browser with an option to either print the report or save the report to a Portable Document Format file. When printing a report, the end-user has the ability to either print the currently displayed form or print a report that closely resembles the existing MPR format. Both of these formats are displayed in Appendix B. When a user with elevated permissions logs into the front-end graphical user interface, the system would present the user with the option of generating MPR records or adding, modifying, or updating existing data within the training database-management system.



Generated reports have PDF export and print capability.

Figure 2. Report Generation

THIS PAGE INTENTIONALLY LEFT BLANK

V. RESULTS DISCUSSION

A. DATA EXPORTING

The sample database created in Microsoft Access contained 350,389 records in a 70-megabyte file. The Visual Basic for Applications scripts exported the Access database to individual comma-separated-value files. The comma-separated-value files totaled 17 megabytes.

The first Visual Basic for Applications script, which exported the files from the Access database to 62 separate comma-separated-value files, took 58.28 seconds. The second Visual Basic for Applications script, which processed the generated comma-separated-value files and formatted the files for proper Oracle formatting, took 142.91 seconds to complete. The total number of files remained at 62. Neither of these scripts generated any errors and had full data accuracy.

B. DATA IMPORTING

Out of the 62 generated comma-separated-value files, 12 were empty and 50 contained data. We analyzed the 12 Access database tables that generated the empty comma-separated-value files and confirmed those tables contained no data. We imported the 50 files into the staging schema using SQL*Loader. When executing the individual SQL*Loader scripts, we initially received errors when importing a file containing an "hours" field. This field was an expected integer, however, end-users accidentally transposed controller-operating initials and hours several times within the database. We confirmed this within the sample database. We were able to enter the records by converting the "hours" field to a string format and processing it later. We additionally received errors due to rows within files containing no data, which we confirmed as well within the sample database. These empty row errors did not prevent the rest of the file from being properly processed. Once we successfully tested our individual scripts, we created a single master script for batch processing. The total execution time for the single master SQL*Loader script was 23.79 seconds.

Our developed production schema consisted of 3.44 megabytes. Our staging schema, once filled with the sample data using SQL*Loader, contained 19.025 megabytes.

C. FRONT-END GRAPHICAL USER INTERFACE TESTING

When creating test data for our production schema, we generated five megabytes worth of sample data for processing. This data was focused on report qualifications, different units, and establishing a proof of concept for generating a new MPR. The data had no correlation to any sample data and we used the data to verify proper record validation.

We tested generating reports for individual members and re-creating hypothetical MPRs. The front-end system successfully generated the required records to re-create a member's MPR. We tested adding additional records and attempting to add invalid records to the system. The system successfully added the records to the member and prevented invalid record entry. After adding the additional record entry, the system properly updated the information in the individual's MPR.

VI. CONCLUSIONS AND FUTURE WORK

A. CONCLUSIONS

In this thesis, we developed a prototype training-management system that processed the data of an existing database that used outdated software and successfully ported it to a modern Oracle database schema. The designed training-management system supports broad implementation throughout the air-traffic-control community and provides flexibility by using both a normalized schema and a separate schema storing the original raw data for every unit. Storing every unit's raw data increases the size of the system, but enables better data validation for every normalized entry within the production schema. After full data validation after importation, we are able to remove the duplicated raw data from the system. The developed system isolates a unit's imported raw data from the production schema, reducing removal complexity and eliminating dependency issues.

A disadvantage of our system is that it requires a significant amount of time to analyze every unit's existing training-management system. This effort requires replication for every unit, as the probability that the same migration process will work for different units is low. Every exportation script will have general similarities, but the scripts are unlikely to be identical unless units are using the same system, platform, and schema. Each unit's script complexity level will vary based on their training system. Units currently using training-management systems based in SQL have a lower complexity and can move directly to data processing and importation.

Once we develop importing tools for each unit, any changes to how a unit stores their data would invalidate the developed scripts. As a result, individual units will be unable to modify their training systems after script development. We do not foresee this happening, but there is a possibility. Finally, units would be required to use both the developed and current training-management systems until full migration is complete. This creates some duplication of effort, but ensures data integrity until any importation or usage issues are resolved.

B. FUTURE WORK

The main goal for this thesis was increasing reliability, accessibility, and data integration for training records. Future work could add capabilities such as offline data entry, automated forwarding to a Military Network Operations Center, integration with the Navy's Visual Information Display System, and refining the front-end graphical user interface.

Offline data entry would allow system operation with limited or no Internet capability, with data uploaded from collection machines once connectivity was re-established. Implemented services, such as Oracle Mobile Sync, would enable tactical field access. We can additionally develop a mobile Application Express application using Oracle's Mobile Application Framework. Having a mobile Application Express application allows users to enter and analyze data from mobile platforms such as tablets or cell phones. Coupled with Oracle's Mobile Sync service, these applications would enable online data access, eliminating data importation requirements in a deployed environment.

Migrating our design to a Military Network Operations Center enables additional features such as Common Access Card login and additional platform security due to network consolidation and enforcing integrated network security architectures. Initial account setup would require establishing a user account and then associating a Common Access Card with that account, but this protocol already exists for several .MIL websites.

Integration with Navy's Visual Information Display System would allow us to import the detailed training data directly to the developed training-management system. This would allow accurate tracking of hours trained on position and would eliminate the need to develop a separate import system for each unit. Currently, there is manual data validation for each record entry, which increases data integration time, but Training Chiefs can more easily identify improper data based on resident-unit knowledge due to the additional oversight. New policy rules are required to replicate this capability, as Visual Information Display System synchronization would only work if units were diligent about accurately managing data entry within the Visual Information Display System.

The front-end graphical user interface only allows basic modification and report generation. We can develop additional report queries to give end-users the ability to generate reports in a view-only mode. Rather than trying to predict every possible use of the system, we can give end-users flexibility in viewing data that may assist them in training or skill development. Currently, any end-user modification requires accessing the back-end and modifying the system data directly. We can add user information options to modify current user permissions and update user information. Additionally, we can implement features to add and drop members within the interface to streamline transferring members between units.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A. EXAMPLE SQL*LOADER FILES

These SQL*Loader files were used to export the POSITION_LOGS table within the Microsoft Access database. We removed the table structures within the control file (line 10) and .LOG output file (line 27) for brevity. The batch file executes the SQL*Loader command, with the batch file specifying the control, .log, and .bad files. The "skip=1" command in line 2 of the batch file skips the first row of the CSV file due to it being a header row.

Line 37 in the .LOG file indicated an error occurred while importing raw data. Lines 30-33 described the error, which indicated a non-integer imported into an integer field. Line 1 of the .BAD file produced the row raising the exception. We identified that a user incorrectly entered a text value in an integer field in the Access database. This entry did not raise an exception in the Access database, but SQL*Loader recognized the error when importing the raw data.

Lines 35-53 of the .LOG file stated the command performance characteristics, which included CPU time, total records processed, and total errors processed.

A. .CTL CONTROL FILE

```
1   load data
2   infile '/CSV_exports/Position_Logs.csv' "str '\r\n'"
3   append
4   into table Position_logs_RAW
5   fields terminated by ','
6   OPTIONALLY ENCLOSED BY '"' AND '"'
7   trailing nullcols
8              (
9
10             /*  table structure */
11
12             )
```

B. .SH BATCH FILE

```
SQLLDR 'username/DB' CONTROL= Position_Logs.ctl LOG=/Logs/
Position_Logs.log BAD=/Logs/ Position_Logs.bad skip=1
```

C. LOG OUTPUT FILES

1. .BAD Output File

```
2/3/2009 0:00,RS,,,,,0,0,,
```

2. .LOG Output File

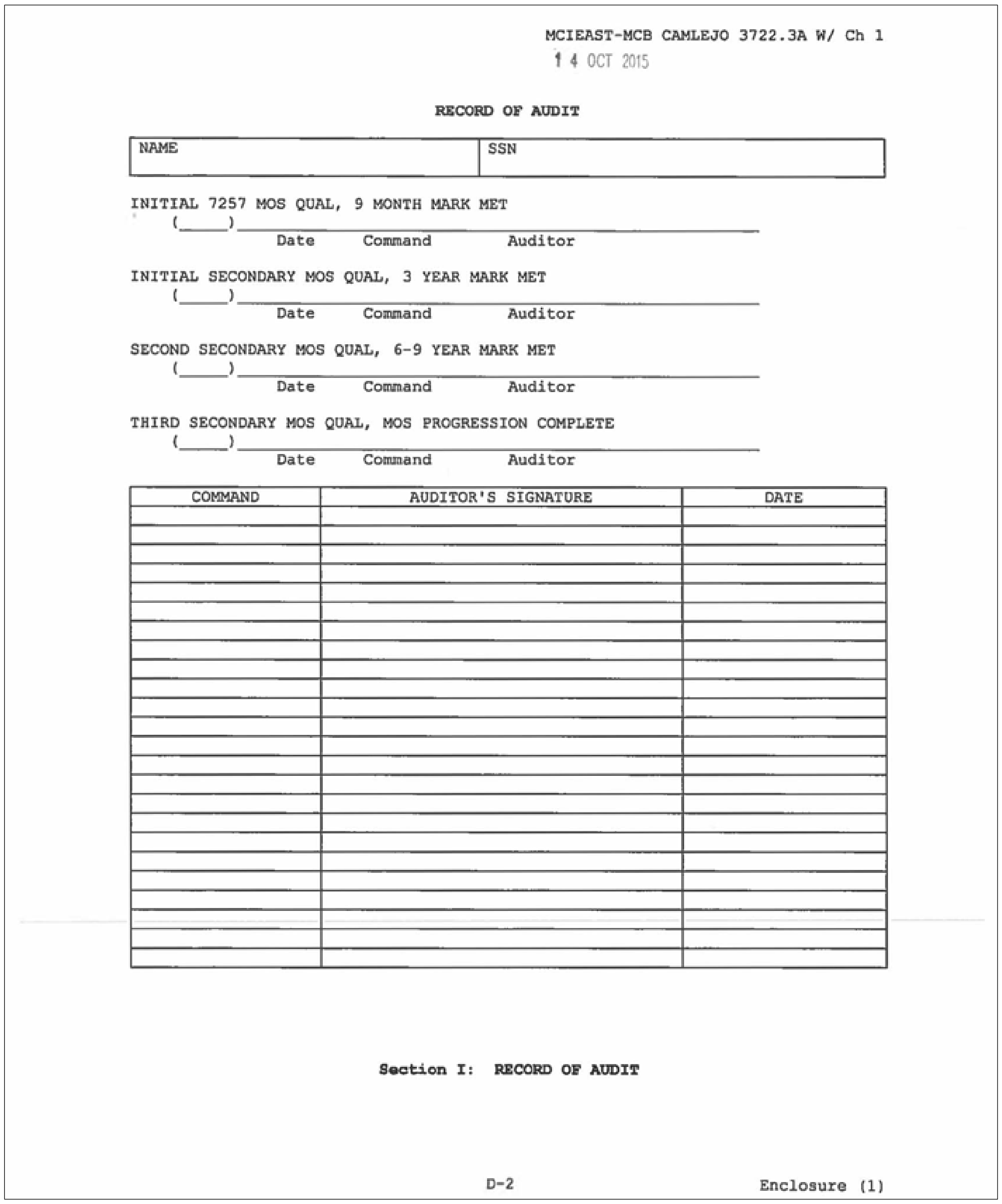
```
SQL*Loader: Release 12.1.0.2.0 - Production on Thu Jul 14 09:18:56 2016
Copyright (c) 1982, 2014, Oracle and/or its affiliates.  All rights reserved.

Control File:   Position_Logs.ctl
Data File:      /CSV_exports/Position_Logs.csv
  File processing option string: "str '
'"
  Bad File:     /Logs/Position_Logs.bad
  Discard File:  none specified

 (Allow all discards)

  Number to load: ALL
  Number to skip: 1
  Errors allowed: 50
  Bind array:     64 rows, maximum of 256000 bytes
  Continuation:    none specified
  Path used:      Conventional

  Table POSITION_LOGS_RAW, loaded from every logical record.
  Insert option in effect for this table: APPEND
  TRAILING NULLCOLS option in effect

     Column Name                  Position   Len  Term Encl Datatype
  ------------------------------ ---------- ----- ---- ---- ---------------------

  /*  table structure */


  value used for ROWS parameter changed from 64 to 6
   Record 223441: Rejected - Error on table POSITION_LOGS_RAW, column
  POSITION.
  ORA-01722: invalid number

  Table POSITION_LOGS_RAW:
```

```
  340423 Rows successfully loaded.
  1 Row not loaded due to data errors.
  0 Rows not loaded because all WHEN clauses were failed.
  0 Rows not loaded because all fields were null.

Space allocated for bind array:                240120 bytes(6 rows)
Read   buffer bytes: 1048576

Total logical records skipped:          1
Total logical records read:        340424
Total logical records rejected:         1
Total logical records discarded:        0

Run began on Thu Jul 14 09:18:56 2016
Run ended on Thu Jul 14 09:21:40 2016

Elapsed time was:     00:02:43.92
CPU time was:         00:00:19.78
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B. ORIGINAL AND GENERATED MPR DOCUMENTS

Figure 3 displays one of the current MPR reports manually completed by the Training Chief. Figure 4 displays how the report appears to a user from the front-end graphical user interface.



MCIEAST-MCB CAMLEJO 3722.3A W/ Ch 1
14 OCT 2015

RECORD OF AUDIT

NAME	SSN

INITIAL 7257 MOS QUAL, 9 MONTH MARK MET
(_____)______________________________
Date Command Auditor

INITIAL SECONDARY MOS QUAL, 3 YEAR MARK MET
(_____)______________________________
Date Command Auditor

SECOND SECONDARY MOS QUAL, 6-9 YEAR MARK MET
(_____)______________________________
Date Command Auditor

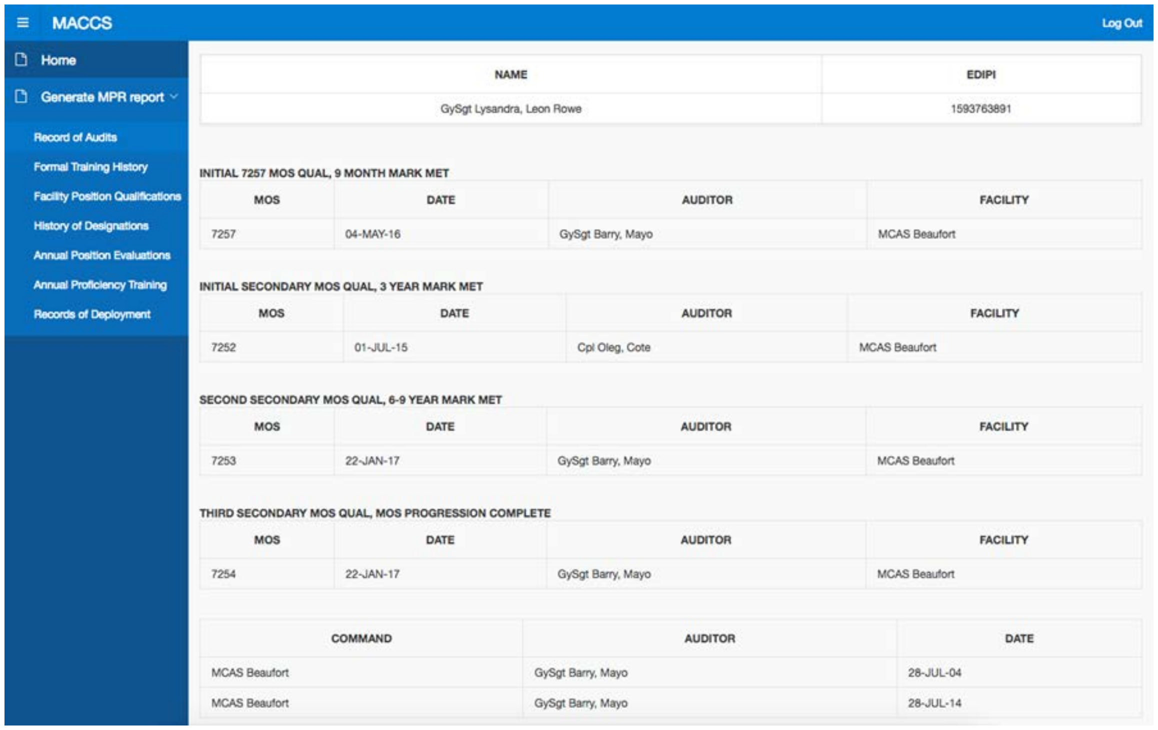
THIRD SECONDARY MOS QUAL, MOS PROGRESSION COMPLETE
(_____)______________________________
Date Command Auditor

COMMAND	AUDITOR'S SIGNATURE	DATE

Section I: RECORD OF AUDIT

D-2 Enclosure (1)

Figure 3. Original MPR Audit Report Record. Source: Department of Defense (2016).



NAME	EDIPI
GySgt Lysandra, Leon Rowe	1593763891

INITIAL 7257 MOS QUAL, 9 MONTH MARK MET

MOS	DATE	AUDITOR	FACILITY
7257	04-MAY-16	GySgt Barry, Mayo	MCAS Beaufort

INITIAL SECONDARY MOS QUAL, 3 YEAR MARK MET

MOS	DATE	AUDITOR	FACILITY
7252	01-JUL-15	Cpl Oleg, Cote	MCAS Beaufort

SECOND SECONDARY MOS QUAL, 6-9 YEAR MARK MET

MOS	DATE	AUDITOR	FACILITY
7253	22-JAN-17	GySgt Barry, Mayo	MCAS Beaufort

THIRD SECONDARY MOS QUAL, MOS PROGRESSION COMPLETE

MOS	DATE	AUDITOR	FACILITY
7254	22-JAN-17	GySgt Barry, Mayo	MCAS Beaufort

COMMAND	AUDITOR	DATE
MCAS Beaufort	GySgt Barry, Mayo	28-JUL-04
MCAS Beaufort	GySgt Barry, Mayo	28-JUL-14

Figure 4. Front-End Display of Generated MPR Audit Report Record

LIST OF REFERENCES

Blischak, J. D., Davenport, E. R., & Wilson, G. (2016). A quick introduction to version control with Git and GitHub. *PLOS Computational Biology*, *12*(1). doi:10.1371/journal.pcbi.1004668

Cargill, C. F. (2011). Why standardization efforts fail. *The Journal of Electronic Publishing*, *14*(1). doi:10.3998/3336451.0014.103

Chief of Naval Operations. (2002). *NATOPS air traffic control manual* (NAVAIR 00-80T-114). Washington, DC: Department of the Navy.

Department of Defense. (2004). *MOS skill designations for Marine air traffic controllers* (230/04). Washington, DC: Author.

Department of Defense. (2008). *DoD operations security (OPSEC) program manual* (5205.02-M). Washington, DC: Author.

Department of Defense. (2012). *Reduction of social security number (SSN) use within DoD* (1000.30). Washington, DC: Author.

Department of Defense. (2016). *Marine Corps Installations East-Marine Corps Base Camp Lejeune air traffic control order (short title: MCIEAST-MCB CAMELJ ATC ORDER)* (3722.3A). Washington, DC: Author.

Department of the Navy. (2011). *Aviation training and readiness program manual* (3500.14C). Washington, DC: Author.

Department of the Navy. (2012a). *Marine air traffic control (MATC) tactical user training and readiness (T&R) manual* (3500.94). Washington, DC: Author.

Department of the Navy. (2012b). *Marine Corps cybersecurity program (MCCSP)* (5239.2A). Washington, DC: Author.

Department of the Navy. (2013). *Military occupational specialties manual (short title: MOS manual)* (1200.17E). Washington, DC: Author.

Department of the Navy. (2016). *Aviation training and readiness program manual* (3500.14D). Washington, DC: Author.

Dryden, P. (1998). Custom tools pose problems. *Computerworld*, *32*(24), 49, 54.

Haerder, T., & Reuter, A. (1983). Principles of transaction-oriented database recovery. *Computing Surveys*, *15*(4), 287–317. doi:10.1145/289.291

InnovaSystems International. (2016). *M-SHARP software user manual* (1.0.64.1). Washington, DC: Author.

Lee, H., Chapiro, J., Schernthaner, R. E., Duran, R., Wang, Z., Gorodetski, B., … Lin, M. (2015). How I do it: A practical database management system to assist clinical research teams with data collecting, organization, and reporting. *Acad Radiol*, *22*(4), 527–533. doi:10.1016/j.acra.2014.12.002

Münstermann, B., Weitzel, T., & Eckhardt, A. (2010). The performance impact of business process standardization. *Business Process Management Journal*, *16*(1), 29–56. doi:10.1108/01409171011070332

Oracle. (2002). SQL*Loader control file reference. Retrieved from http://docs.oracle.com/cd/A97630_01/server.920/a96652/ch05.htm

Oracle. (2016). Developer day - hands-on database application development. Retrieved from http://www.oracle.com/technetwork/database/enterprise-edition/databaseappdev-vm-161299.html

Oracle. (2017). Schema object names and qualifiers. Retrieved from https://docs.oracle.com/cd/B19306_01/server.102/b14200/sql_elements008.htm

Pavlo, A., Paulson, E., Rasin, A., Abadi, D. J., DeWitt, D. J., Madden, S., & Stonebraker, M. (2009). *A comparison of approaches to large-scale data analysis*. *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data*. New York, NY: ACM. doi:10.1145/1559845.1559865

Portfolio, T., Godwin, J., Arnold, S., Dreskin, S., Dufour, P., Faris, S., … Vasterd, P. (1996). Programmer’s guide to the Oracle precompilers. *Oracle*, (February).

Siha, S. M., & Saad, G. H. (2008). Business process improvement: Empirical assessment and extensions. *Business Process Management Journal*, *14*(6), 778–802. doi:10.1108/14637150810915973

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California